
Siren: a hypermedia specification for representing entities

DOI [10.5281/zenodo.556783](https://doi.org/10.5281/zenodo.556783)

Your input is appreciated. Feel free to file a GitHub Issue, a Pull Request, or contact us. Thank you!

- Official Siren Google Group
- Kevin on Twitter @kevinswiber

Example

Below is a JSON Siren example of an order, including sub-entities. The first sub-entity, a collection of items associated with the order, is an embedded link. Clients may choose to automatically resolve linked sub-entities. The second sub-entity is an embedded representation of customer information associated with the order. The example also includes an action to add items to the order and a set of links to navigate through a list of orders.

The media type for JSON Siren is `application/vnd.siren+json`.

```
1 {
2   "class": [ "order" ],
3   "properties": {
4     "orderNumber": 42,
5     "itemCount": 3,
6     "status": "pending"
7   },
8   "entities": [
9     {
10      "class": [ "items", "collection" ],
11      "rel": [ "http://x.io/rels/order-items" ],
12      "href": "http://api.x.io/orders/42/items"
13    },
14    {
15      "class": [ "info", "customer" ],
16      "rel": [ "http://x.io/rels/customer" ],
17      "properties": {
18        "customerId": "pj123",
19        "name": "Peter Joseph"
20      },
21      "links": [
22        { "rel": [ "self" ], "href": "http://api.x.io/customers/pj123"
23        }
24      ]
25    },
26    "actions": [
27      {
```

```
28     "name": "add-item",
29     "title": "Add Item",
30     "method": "POST",
31     "href": "http://api.x.io/orders/42/items",
32     "type": "application/x-www-form-urlencoded",
33     "fields": [
34         { "name": "orderNumber", "type": "hidden", "value": "42" },
35         { "name": "productCode", "type": "text" },
36         { "name": "quantity", "type": "number" }
37     ]
38 }
39 ],
40 "links": [
41     { "rel": [ "self" ], "href": "http://api.x.io/orders/42" },
42     { "rel": [ "previous" ], "href": "http://api.x.io/orders/41" },
43     { "rel": [ "next" ], "href": "http://api.x.io/orders/43" }
44 ]
45 }
```

Introduction

Siren is a hypermedia specification for representing entities. As HTML is used for visually representing documents on a Web site, Siren is a specification for presenting entities via a Web API. Siren offers structures to communicate information about entities, actions for executing state transitions, and links for client navigation.

Siren is intended to be a general specification of a generic media type that can be applied to other types that are not inherently hypermedia-powered. The initial implementation is JSON Siren. Other implementations, such as XML Siren, may also be implemented using the Siren specification.

All examples in this document are in the JSON Siren format.

Entities

An Entity is a URI-addressable resource that has properties and actions associated with it. It may contain sub-entities and navigational links.

Root entities and sub-entities that are embedded representations SHOULD contain a `links` collection with at least one item contain a `rel` value of `self` and an `href` attribute with a value of the entity's URI.

Sub-entities that are embedded links MUST contain an `href` attribute with a value of its URI.

Entity

class Describes the nature of an entity's content based on the current representation. Possible values are implementation-dependent and should be documented. MUST be an array of strings. Optional.

properties A set of key-value pairs that describe the state of an entity. In JSON Siren, this is an object such as { "name": "Kevin", "age": 30 }. Optional.

entities A collection of related sub-entities. If a sub-entity contains an href value, it should be treated as an embedded link. Clients may choose to optimistically load embedded links. If no href value exists, the sub-entity is an embedded entity representation that contains all the characteristics of a typical entity. One difference is that a sub-entity MUST contain a rel attribute to describe its relationship to the parent entity.

In JSON Siren, this is represented as an array. Optional.

links A collection of items that describe navigational links, distinct from entity relationships. Link items should contain a rel attribute to describe the relationship and an href attribute to point to the target URI. Entities should include a link rel to self. In JSON Siren, this is represented as "links": [{ "rel": ["self"], "href": "http://api.x.io/orders/1234"}] Optional.

actions A collection of action objects, represented in JSON Siren as an array such as { "actions": [{ ... }] }. See Actions. Optional

title Descriptive text about the entity. Optional.

Sub-Entities

Sub-entities can be expressed as either an embedded link or an embedded representation. In JSON Siren, sub-entities are represented by an entities array, such as { "entities": [{ ... }] }.

Embedded Link A sub-entity that's an embedded link may contain the following:

class Describes the nature of an entity's content based on the current representation. Possible values are implementation-dependent and should be documented. MUST be an array of strings. Optional.

rel Defines the relationship of the sub-entity to its parent, per Web Linking (RFC5988) and Link Relations. MUST be a non-empty array of strings. Required.

href The URI of the linked sub-entity. Required.

type Defines media type of the linked sub-entity, per Web Linking (RFC5988). Optional.

title Descriptive text about the entity. Optional.

Embedded Representation Embedded sub-entity representations retain all the characteristics of a standard entity, but MUST also contain a **rel** attribute describing the relationship of the sub-entity to its parent.

Classes vs. Relationships

It's important to note the distinction between link relations and classes. Link relations define a relationship between two resources. Classes define a classification of the nature of the element, be it an entity or an action, in its current representation.

Sub-Entities vs Links

Another distinction is the difference between sub-entities and links. Sub-entities exist to communicate a relationship between entities, in context. Links are primarily navigational and communicate ways clients can navigate outside the entity graph.

Links

Links represent navigational transitions. In JSON Siren, links are represented as an array inside the entity, such as { **"links"**: [{ **"rel"**: [**"self"**], **"href"**: **"http://api.x.io/orders/42"**] } }

Links may contain the following attributes:

rel

Defines the relationship of the link to its entity, per Web Linking (RFC5988) and Link Relations. MUST be an array of strings. Required.

class

Describes aspects of the link based on the current representation. Possible values are implementation-dependent and should be documented. MUST be an array of strings. Optional.

href

The URI of the linked resource. Required.

title

Text describing the nature of a link. Optional.

type

Defines media type of the linked resource, per Web Linking (RFC5988). Optional.

Actions

Actions show available behaviors an entity exposes.

name

A string that identifies the action to be performed. Action names MUST be unique within the set of actions for an entity. The behaviour of clients when parsing a Siren document that violates this constraint is undefined. Required.

class

Describes the nature of an action based on the current representation. Possible values are implementation-dependent and should be documented. MUST be an array of strings. Optional.

method

An enumerated attribute mapping to a protocol method. For HTTP, these values may be [GET](#), [PUT](#), [POST](#), [DELETE](#), or [PATCH](#). As new methods are introduced, this list can be extended. If this attribute is omitted, [GET](#) should be assumed. Optional.

href

The URI of the action. Required.

title

Descriptive text about the action. Optional.

type

The encoding type for the request. When omitted and the [fields](#) attribute exists, the default value is [application/x-www-form-urlencoded](#). Optional.

fields

A collection of fields, expressed as an array of objects in JSON Siren such as { ["fields"](#): [{ ... }] }. See Fields. Optional.

Fields

Fields represent controls inside of actions. They may contain these attributes:

name A name describing the control. Field names **MUST** be unique within the set of fields for an action. The behaviour of clients when parsing a Siren document that violates this constraint is undefined. Required.

class

Describes aspects of the field based on the current representation. Possible values are implementation-dependent and should be documented. **MUST** be an array of strings. Optional.

type The input type of the field. This may include any of the following input types specified in HTML5:

`hidden`, `text`, `search`, `tel`, `url`, `email`, `password`, `datetime`, `date`, `month`, `week`, `time`, `datetime-local`, `number`, `range`, `color`, `checkbox`, `radio`, `file`

When missing, the default value is `text`. Serialization of these fields will depend on the value of the action's `type` attribute. See `type` under Actions, above. Optional.

value A value assigned to the field. Optional.

title Textual annotation of a field. Clients may use this as a label. Optional.

Usage Considerations

Siren supports a resource design style that doesn't have to be primarily CRUD-based. A root entity may take ownership of facilitating changes to sub-entities via actions. Using Siren allows you to easily provide a task-based interface through your Web API.

Feedback

Siren is still a work in progress looking for some real world usage and feedback. Please contribute! Feel free to use GitHub Issues to make suggestions or fire up a Pull Request with changes to be reviewed. Thank you!